# 韩山师范学院实验报告

姓名：林婷婷　　专业：信息管理与信息系统　班级：20181171　　学号：2018117126

科目：Android 应用开发　　实验日期：2021.6.10

实验题目：

【实验目的】

UI 开发工具编写程序界面

【实验环境及方式】

Android Studio

【实验内容及实验结果】

**常用控件的使用方法**

1. **Text View**

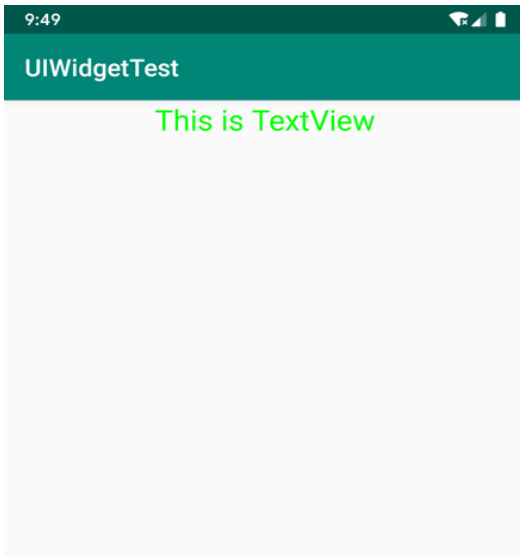
TextView 主要用于在界面上显示一段文本信息。

```
<TextView
    android:id="@+id/textView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:textColor="#00ff00"
    android:textSize="24sp"
    android:text="This is TextView"/>
```

使用 andriod:gravity 来指定文字对齐方式，可选值有 top、bottom、start、end、center 等，我们使用 center 来表示文字在垂直和水平方向都居中对齐。

通过 android:textColor 属性指定文字颜色，通过 android:textSize 属性改变字体大小，字体大小使用 sp 作为单位。

效果如图所示。

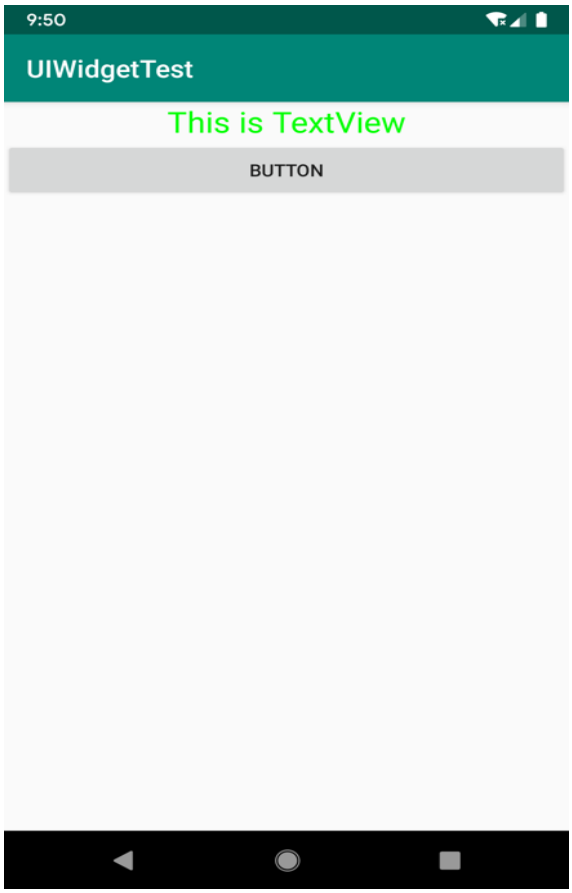


2. **Button**

Button 是程序用于和用户进行交互的一个重要控件。

```
<Button
    android:id="@+id/button"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Button" />

    button.setOnClickListener {
        // 在此处添加逻辑
    }
```

加入 button 后的界面如图所示



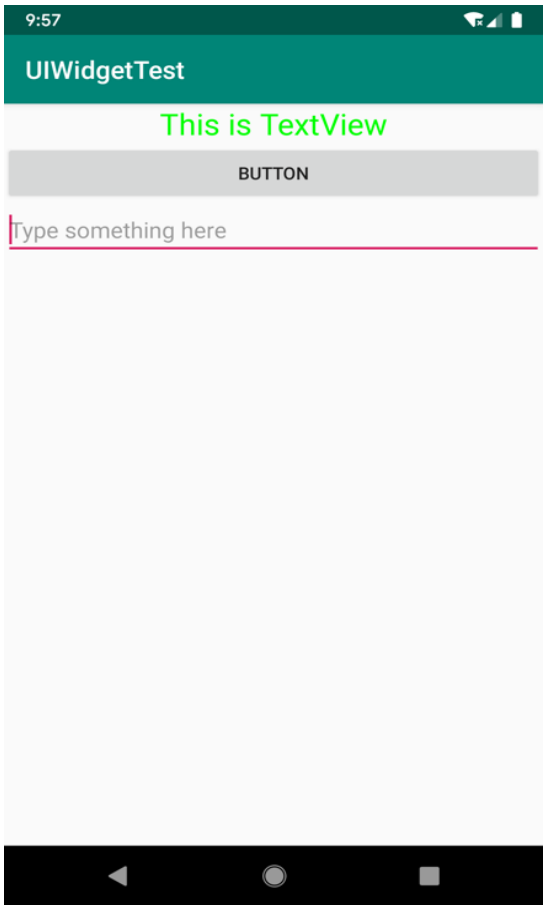

这里调用 button 的 setOnClickListener( )方法时利用了 Java 单抽象方法接口的特性。

3. **Edit Text**

Edit Text 是程序用于和用户进行交互的另一个重要控件，它允许用户在控件里输入和编辑内容，并可以在程序中对这些内容进行处理。

```
<EditText
    android:id="@+id/editText"
    android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content"
    android:hint="Type something here"
    />
```

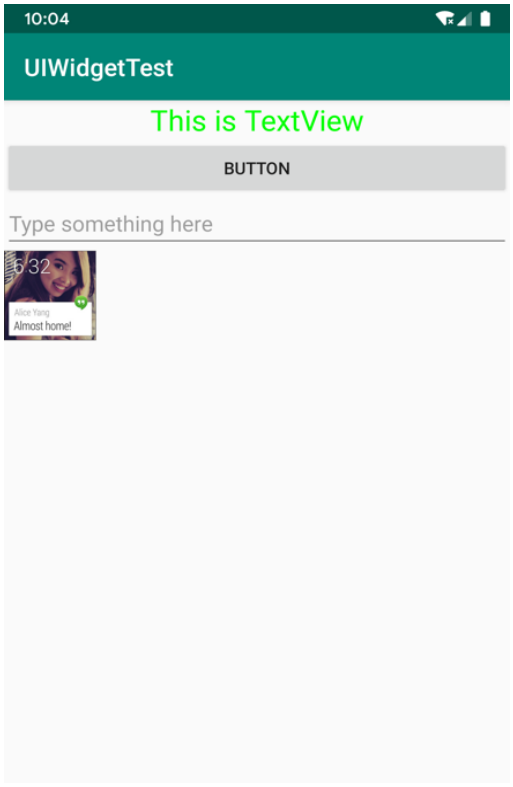



4. **Image View**

Image View 是用于在界面上展示图片的一个控件，它可以让我们的程序界面变得更加丰富多彩。图片通常是放在以 drawable 开头的目录下，并且要带上具体的分辨率，现在主流手机屏幕分辨率大多是 xxhdpi 的，所以我们在 res 目录下再新建一个 drawable-xxhdpi 目录然后将事先准备好的图片复制进去。

接着修改 activity_main.xml，如下所示：

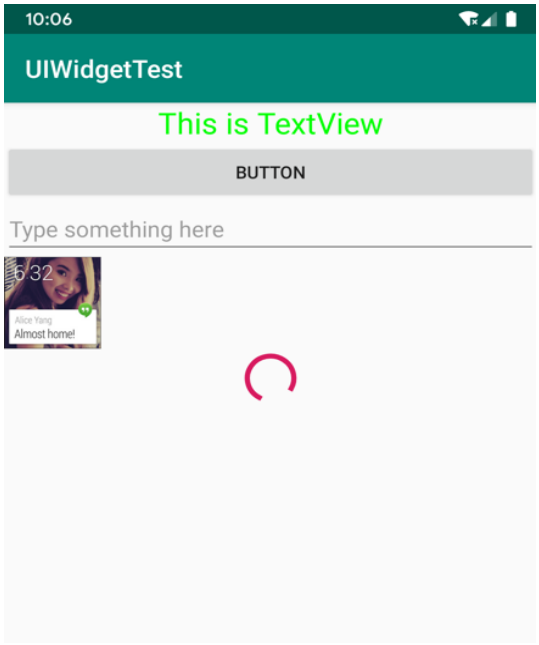
```
<ImageView
    android:id="@+id/imageView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/img_1"
    />
```




5. **Progress Bar**

Progress Bar 用于在界面上显示一个进度条，表示我们的程序正在加载一些数据。

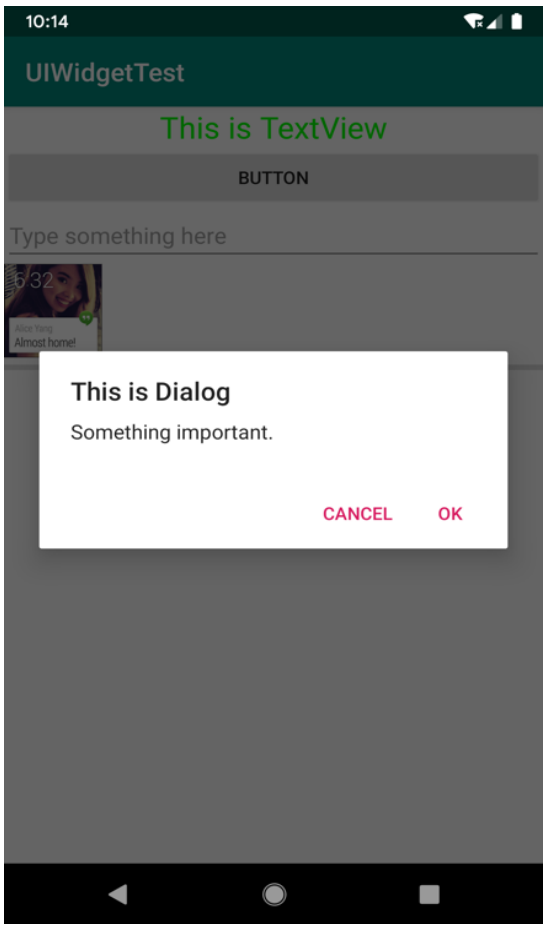
```
<ProgressBar
    android:id="@+id/progressBar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    />
```




6. **Alert Dialog**

Alert Dialog 可以在当前界面弹出一个对话框，这个对话框是置顶于所有界面元素之上的，能够屏蔽其他控件的交互能力，因此 AlertDialog 一般用于提示一些非常重要的内容或者警告信息。

```
AlertDialog.Builder(this).apply {
        setTitle("This is Dialog")
        setMessage("Something important.")
        setCancelable(false)
        setPositiveButton("OK") { dialog, which ->
        }
        setNegativeButton("Cancel") { dialog, which ->
        }
        show()
        }
```




3 种基本布局

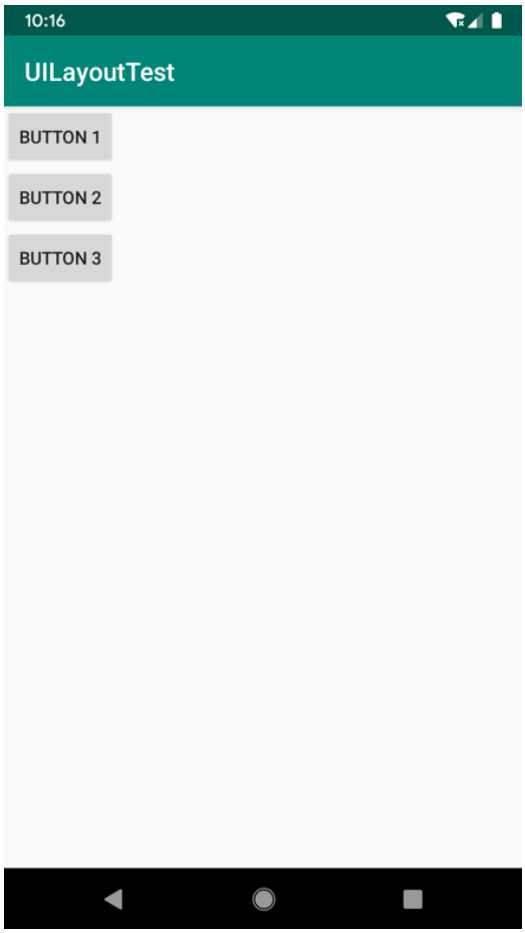
7. **Linear Layout**

Linear Layout 又称作线性布局，这个布局会将它所包含的控件在线性方向上依次排列。

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical"
android:layout_width="match_parent"
android:layout_height="match_parent">
```

```
<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Button 1" />

<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Button 2" />

<Button
    android:id="@+id/button3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Button 3" />

</LinearLayout>
```

显示结果如图所示：



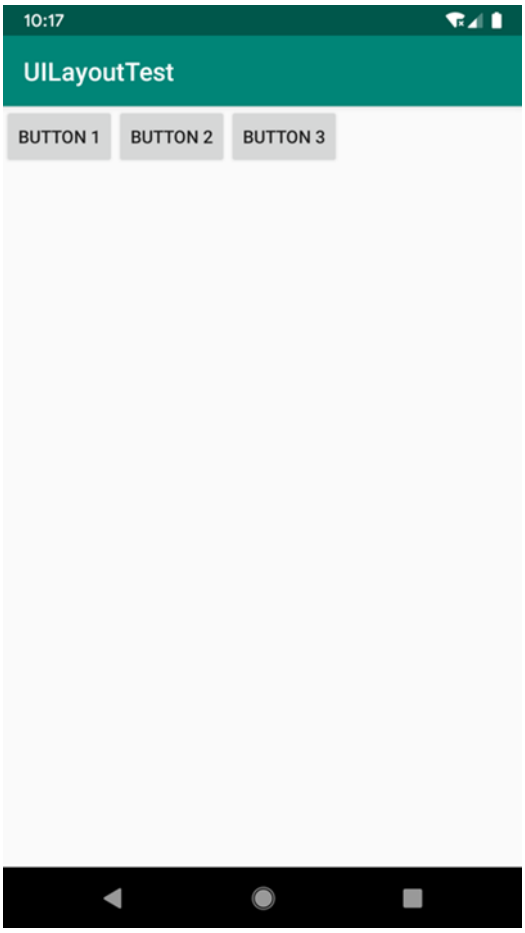

LinearLayout 的排列方式也可改变，将 andriod:orientation 属性的值改为 horizontal，变成横向排列的效果。

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="horizontal"

android:layout_width="match_parent"

android:layout_height="match_parent">




LinearLayout 使用 ayout_gravity 属性对齐的效果：

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="top"
        android:text="Button 1" />
    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_vertical"
        android:text="Button 2" />
```

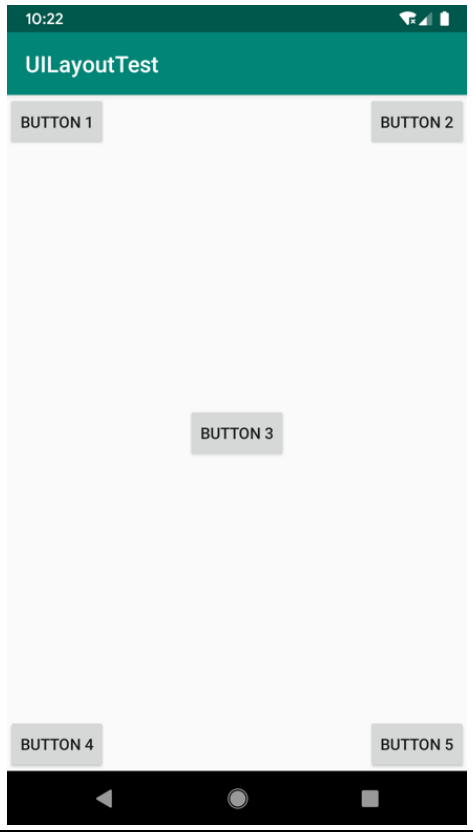
```
    <Button
        android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom"
        android:text="Button 3" />
    </LinearLayout>
```


8. **Relative Layout**

Relative Layout 又称作相对布局，它可以通过相对定位的方式让控件出现在布局的任何位置，更为随意。

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout _width=”match_parent”
    android:layout _height=” match_parent”>
<Button
android:id=”@+id/button1”
    android:layout _width=”wrap_content”
    android:layout _height=” wrap_ content”
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:text="Button 1" />
<Button
android:id=”@+id/button2”
```

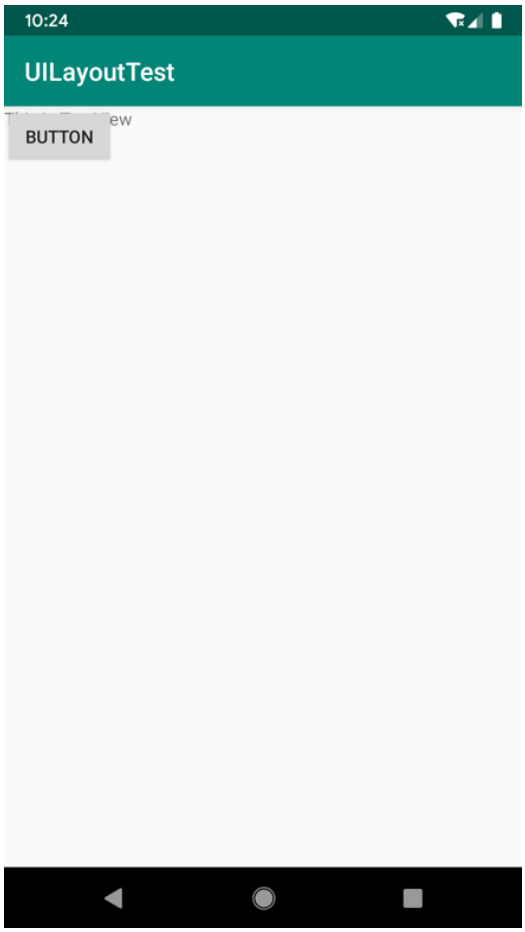
```
    android:layout _width=”wrap_ content”
    android:layout _height=” wrap_ content”
    android:layout_alignParentRight="true"
    android:layout_alignParentTop="true"
    android:text="Button 2" />
<Button
android:id=”@+id/button3”
    android:layout _width=”wrap_ content”
    android:layout _height=” wrap_ content”
    android:layout_centerInParent="true"
    android:text="Button 3" />
<Button
android:id=”@+id/button4”
    android:layout _width=”wrap_ content”
    android:layout _height=” wrap_ content”
    android:layout_alignParentBottom="true"
    android:layout_alignParentLeft="true"
    android:text="Button 4" />
<Button
android:id=”@+id/button5”
    android:layout _width=”wrap_ content”
    android:layout _height=” wrap_ content”
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:text="Button 5" />
    </RelativeLayout>
```



9. **FrameLayout**

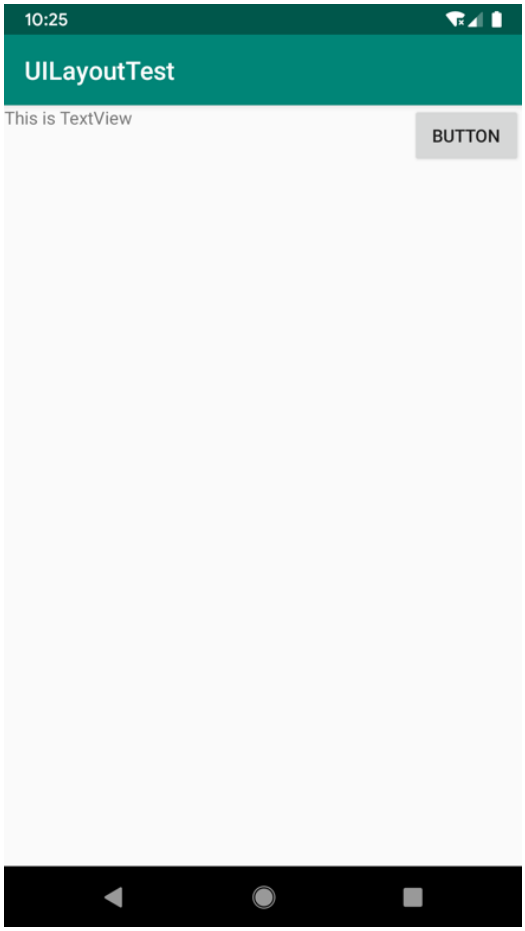
FrameLayout 又称作帧布局，这种布局没有丰富的定位方式，所有的控件都会默认摆放在布局的左上角。

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="This is TextView"
        />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        />

</FrameLayout>
```



除了默认效果之外，还可以使用 layout_gravity 属性来指定控件在布局中的对齐方式，这和 LinearLayout 中的用法是相似的。

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="left"
        android:text="This is TextView"
        />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="right"
        android:text="Button"
        />
</FrameLayout>
```



10. **Recycler View**

① 添加 RecyclerView 控件

在布局中加入 RecyclerView 控件也是非常简单的，先为 RecyclerView 指定一个 id，然后将宽度和高度都设置为 match_parent，这样 RecyclerView 就占满了整个布局的空间。

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerView"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />

</LinearLayout>
```

② 定义实体类和子项布局

接着定义一个实体类，作为 RecyclerView 适配器的适配类型。

新建 Fruit 类，代码如下所示：class Fruit(val name:String, val imageId: Int)

③ 然后需要为 RecyclerView 的子项指定一个我们自定义的布局，在 layout 目录下新建 fruit_item.xml，代码如下所示：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="60dp">

    <ImageView
        android:id="@+id/fruitImage"
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:layout_gravity="center_vertical"
        android:layout_marginLeft="10dp"/>

    <TextView
        android:id="@+id/fruitName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_vertical"
        android:layout_marginLeft="10dp" />

</LinearLayout>
```

④ 为 RecyclerView 准备一个适配器，新建 FruitAdapter 类，让这个适配器继承自 RecyclerView.Adapter，并将泛型指定为 FruitAdapter.ViewHolder。

```
class FruitAdapter(val fruitList: List<Fruit>) :
```

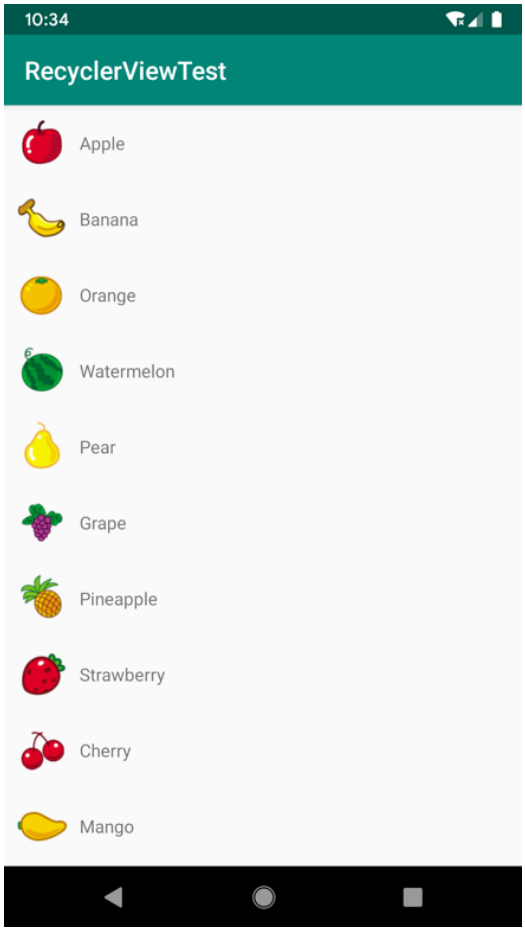
```
RecyclerView.Adapter<FruitAdapter.ViewHolder>() {

inner class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {
val fruitImage: ImageView = view.findViewById(R.id.fruitImage)
val fruitName: TextView = view.findViewById(R.id.fruitName)
}

override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
val view = LayoutInflater.from(parent.context).inflate(R.layout.fruit_item, parent,
false)
return ViewHolder(view)
}

override fun onBindViewHolder(holder: ViewHolder, position: Int) {
val fruit = fruitList[position]
holder.fruitImage.setImageResource(fruit.imageId)
holder.fruitName.text = fruit.name
}

override fun getItemCount() = fruitList.size

}
```

⑤ 适配器准备好之后，可以开始使用 RecyclerView。

```
class MainActivity : AppCompatActivity() {

    private val fruitList = ArrayList<Fruit>()

override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)
initFruits() // 初始化水果数据
val layoutManager = LinearLayoutManager(this)
recyclerView.layoutManager = layoutManager
val adapter = FruitAdapter(fruitList)
recyclerView.adapter = adapter
}

private fun initFruits() {
repeat(2) {
fruitList.add(Fruit("Apple", R.drawable.apple_pic))
fruitList.add(Fruit("Banana", R.drawable.banana_pic))
fruitList.add(Fruit("Orange", R.drawable.orange_pic))
fruitList.add(Fruit("Watermelon", R.drawable.watermelon_pic))
fruitList.add(Fruit("Pear", R.drawable.pear_pic))
fruitList.add(Fruit("Grape", R.drawable.grape_pic))
```

```
fruitList.add(Fruit("Pineapple", R.drawable.pineapple_pic))
fruitList.add(Fruit("Strawberry", R.drawable.strawberry_pic))
fruitList.add(Fruit("Cherry", R.drawable.cherry_pic))
fruitList.add(Fruit("Mango", R.drawable.mango_pic))
}
}

}
```

最后显示如图所示：



【教师评语和成绩】

**成绩：** **指导教师：** **日期：**